



Von den Daten über das Wissen zur Information

Wie funktioniert Data Mining? Oder, wie funktioniert ein Spamfilter? Aus einer Datenbank, in diesem Fall einem Postfach, werden im ersten Schritt Datensätze, hier E-Mails, nach bestimmten Kriterien ausgewählt und, gegebenenfalls manuell, klassifiziert (nach „Spam“ und „kein Spam“). Anschließend werden die Daten gesäubert und in eine für die späteren Berechnungen besser geeignete Darstellung überführt. Im Falle eines Spamfilters, oder einer vergleichbaren Anwendung des Textminings, werden zuerst alle Stoppwörter wie „ist“ oder „und“ entfernt. Für die verbleibenden Wörter wird gezählt, wie oft sie in einem Dokument vorkommen. Schließlich werden die Worthäufigkeiten noch normalisiert.

Erst im Anschluss an die Datensäuberung und -aufbereitung folgt der interessante Teil des Data Minings: das Maschinelle Lernen. Hierbei versucht ein Algorithmus auf Basis von einzelnen Beispielen (den Daten) ein allgemeines Modell (das Wissen) zu erlernen, aus dem wiederum interessante Fakten (die Information) abgeleitet werden können. Ein solches Modell kann ausschließlich beschreibend sein (wie beim deskriptiven Mining) oder wie im Falle des Spamfilters (und wie beim präskriptiven Mining) für die automatische Klassifikation von neuen E-Mails verwendet werden.

Ein Vogel lernt fliegen

Weka als Workbench für das Data Mining unterstützt nicht nur die Datenaufbereitung und die Klassifikation wie sie eben beschrieben wurde. Ebenso werden zahlreiche Methoden aus den Bereichen Regression, Clustering, Association Rule Mining und Attribute Selection angeboten. Da die Software in Java geschrieben ist und einem objektorientierten Entwurf folgt, kann sie auch durch eigene Klassen, zum Beispiel Filter für die Daten oder Klassifizierer, erweitert werden. Sie kann auch als Java-Bibliothek in eine bestehende Anwendung integriert werden. Beides verlangt Grundkenntnisse in der objektorientierten Programmierung mit Java und ein Verständnis des Klassenmodells von Weka. Letzteres ist ausreichend dokumentiert

Im Datenrausch

von Martin Szugat

Praktische Einführung in das Data Mining mit Weka 3.4

Fallende Hardwarepreise machen es möglich: Noch nie war es so einfach, und vor allem so billig, Daten zu speichern und zu sammeln. Doch wird es immer schwieriger, aus den vorhandenen Daten wertvolle Informationen zu gewinnen. Der Mensch alleine kann diese Datenflut schon lange nicht mehr bewältigen. Gefragt sind daher zunehmend automatisierte Verfahren, die das in den Daten versteckte Wissen ans Tageslicht befördern. Hiermit beschäftigen sich das so genannte „Data Mining“ und der vorliegende Beitrag.

Beim Data Mining treffen Methoden aus einer Vielzahl wissenschaftlicher Disziplinen zusammen: Der Statistik und der Wahrscheinlichkeitslehre, aber auch der Informatik, der Physik und sogar der Biologie. Um sie selbst zu implementieren, bedarf es allerdings umfangreichen Wissens. Doch die Anwendung derselben, zumal wenn es sich um Standardverfahren handelt, verlangt ein wenig Erfahrung und die passende Software. Mit dem „Waikato Environment for Knowledge Analysis“, kurz „Weka“ genannt, steht nun eine solche als Open-Source-Anwendung unter [1] zur Verfügung. Weka ist übrigens auch der Name eines neuseeländischen, flugunfähigen Vogels, der sich durch seine besondere Neugier auszeichnet – ein passender Name also. Mehr hierzu und zum Thema „Data Mining“ bietet das gleichnamige Buch von den Weka-Entwicklern (siehe [2] und Textkasten).

Quellcode



Der Quellcode zum Artikel befindet sich auf der beiliegenden CD.

(siehe [3]) und wird außerdem in einem frei von [4] erhältlichen Probekapitel aus dem erwähnten Buch beschrieben.

Dem reinen Anwender präsentiert sich Weka entweder als umfangreiche Sammlung von Konsolenprogrammen (ebenefalls in [4] beschrieben) oder in Form einer graphischen Benutzeroberfläche. Letztere unterteilt sich in drei voneinander unabhängige Anwendungen: dem „Explorer“ für die Datenaufbereitung und -analyse, einem visuellen Designer namens „KnowledgeFlow“ und dem „Experimenter“, um verschiedene Klassifizierer auf unterschiedlichen Datenmengen zu vergleichen. Die Applikation KnowledgeFlow befindet sich noch in der Entwicklung und wird daher hier nicht weiter vorgestellt. Es lohnt sich jedoch, einen Blick darauf zu werfen, da hiermit auf intuitive Weise auch komplizierte Prozessabläufe modelliert werden können.

Sämtliche funktionalen Module von Weka (Filter, Klassifizierer, etc.) sind in allen drei Anwendungen ebenso wie in der Kommandozeile verfügbar. Das verbindende Element ist das Weka-eigene ARFF-Format, obgleich auch für andere Forma-

te wie beispielsweise komma-separierte Dateien entsprechende Konverter zur Verfügung stehen. „ARFF“ steht für „Attribute-Relation File Format“ und bezeichnet ein textbasiertes Format für relationale Daten, welches unter [5] dokumentiert ist. Eine ARFF-Datei kann jeweils nur eine Relation (entspricht einer Tabelle) enthalten, deren Namen in der ersten Zeile der Datei nach dem Schlüsselwort „@relation“ (Groß- und Kleinschreibung spielt hierbei keine Rolle) zu stehen hat. Eine Erweiterung für mehrrelationale Daten steht mit dem MARFF-Format ebenfalls zur Verfügung (siehe [6]).

Das Zeichen „@“ leitet allgemein ein Schlüsselwort ein. Mit „%“ können dagegen Kommentarzeilen eingefügt werden. Ein weiteres Schlüsselwort ist „attribute“, welches ein Attribut (entspricht einer Spalte in einer Tabelle) beschreibt und welches vor den eigentlichen Daten, aber nach dem Namen der Relation stehen muss. Für jedes Attribut wird eine Zeile beginnend mit der Zeichenfolge „@attribute“ hinzugefügt. Durch Leerzeichen getrennt folgen der eindeutige Attributname und der Attributtyp. Erlaubte Typen sind „numeric“ für numerische Attribute, „string“ für beliebige Zeichenfolgen, „date“ für Datumsangaben gemäß der Java-Formatierung und eine in geschweifte Klammern gesetzte Auflistung von möglichen nominalen Werten. Diese werden innerhalb der geschweiften Klammern durch Kommata getrennt.

Schließlich steht das Schlüsselwort „@data“ in einer separaten Zeile. Dieser Zeile folgen die eigentlichen Daten, wobei pro Datensatz, in Weka als Instanz bezeichnet, eine Zeile zu verwenden ist. Gefüllt werden die Zeilen mit den jeweiligen Attributwerten, wobei die Reihenfolge der Werte mit der Reihenfolge in der Deklaration der Attribute übereinstimmen muss. Nur beim Sparse ARFF (zu Deutsch „spärlich“) werden Indizes verwendet. Hierbei werden sämtliche Attribute, denen kein Wert zugewiesen wurde, mit Null belegt.

Für die Trennung der Attributwerte wird erneut das Komma verwendet. Die Abtrennung der Nachkommastellen in Dezimalzahlen erfolgt hingegen – wie im Englischen üblich – mittels des Punktes. Zeichenfolgen sollten zudem in Anführungszeichen eingeschlossen werden, fehlende Werte

werden mit einem Fragezeichen dargestellt und für nominale Attribute wird der konkrete Wert (und nicht etwa ein Index) eingesetzt.

Ein Beispiel für eine ARFF-Datei mit gemischten Attributen (numerisch und nominal) ist in Listing 1 zu sehen. Sie entstammt der Weka-Distribution, die von der genannten Website unter [1] für Windows, Mac und Unix respektive Linux heruntergeladen werden kann. Für Linux steht die ZIP-Datei auch auf der Heft-CD zur Verfügung. Nachdem die Datei entpackt wurde, finden sich im Unterverzeichnis */data* eine Reihe von exemplarischen ARFF-Dateien, darunter die gezeigte Datei *weather.arff*. Sie beschreibt einen Datensatz, der Auskunft darüber gibt, ob bei bestimmten Wetterverhältnissen (Ausblick, Temperatur, Luftfeuchtigkeit und Windigkeit) ein nicht näher genanntes Spiel gespielt werden sollte oder nicht. Hierzu sind einige Beispieldatensätze spezifiziert, anhand derer im Folgenden ein Modell entwickelt werden soll, um für zukünftige Tage zu entscheiden, ob ein Spiel sinnvoll ist oder nicht.

Spieltrieb

Um das beste Modell zu finden, also jenes mit der größten Vorhersagegenauigkeit, werden verschiedene Klassifizierer ausprobiert. Hierfür werden die Klassifizierer mit den gegebenen Daten trainiert und anschließend getestet. Da für das Training und den Test unterschiedliche Datenmengen verwendet werden sollten (um eine zu optimistische Auswertung zu vermeiden), werden die Daten in eine Trainings- und eine Testmenge aufgeteilt, wobei die Testmenge etwa 33 Prozent der Instanzen umfassen sollte.

Alternativ hierzu kann auch eine *k*-fache Kreuzvalidierung durchgeführt werden, wobei die Daten in *k*-gleichgroße Teile zerlegt werden und für das Training *k* - 1 der Teilmengen verwendet werden. Mit dem verbleibenden Teil wird dann getestet. Dieser Vorgang wird *k*-mal wiederholt, wobei jeweils eine andere der *k*-Teilmengen eingesetzt wird. Brauchbare, das heißt statistisch signifikante Ergebnisse, liefert der oftmals gebrauchte Wert 10 für *k*. Ist *k* gleich der Anzahl an Instanzen, so wird dies als Leave-One-Out Kreuzvalidierung bezeichnet.

Listing 1

Die Datei *weather.arff* aus der Weka-Distribution

```
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature real
@attribute humidity real
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
rainy,68,80,FALSE,yes
rainy,65,70,TRUE,no
overcast,64,65,TRUE,yes
sunny,72,95,FALSE,no
sunny,69,70,FALSE,yes
rainy,75,80,FALSE,yes
sunny,75,70,TRUE,yes
overcast,72,90,TRUE,yes
overcast,81,75,FALSE,yes
rainy,71,91,TRUE,no
```

Da die k -Validierungen unabhängig voneinander verlaufen und eine Validierung bei großen Datenmengen oder aufwendigen Verfahren durchaus seine Zeit beansprucht, wäre es vorteilhaft, sie auf verschiedene Rechner zu verteilen. Weka unterstützt dies von Haus aus nicht, aber mit Weka-Parallel (siehe [7]) steht eine auf der ursprünglichen Weka-Version 3.2 basierende Software zur Verfügung, die dies leistet.

Hier sei jedoch nur das Original beschrieben. Um es zu starten, muss die verantwortliche JAR-Datei *weka.jar* mit dem Java-Interpreter zur Ausführung gebracht werden. Gegebenenfalls sollten die maximale Heap-Größe noch heraufgesetzt und die JDBC-Treiber geladen werden:

```
java -Xmx512M -cp $CLASSPATH -jar weka.jar -Ddriver=com.mysql.jdbc.Driver
```

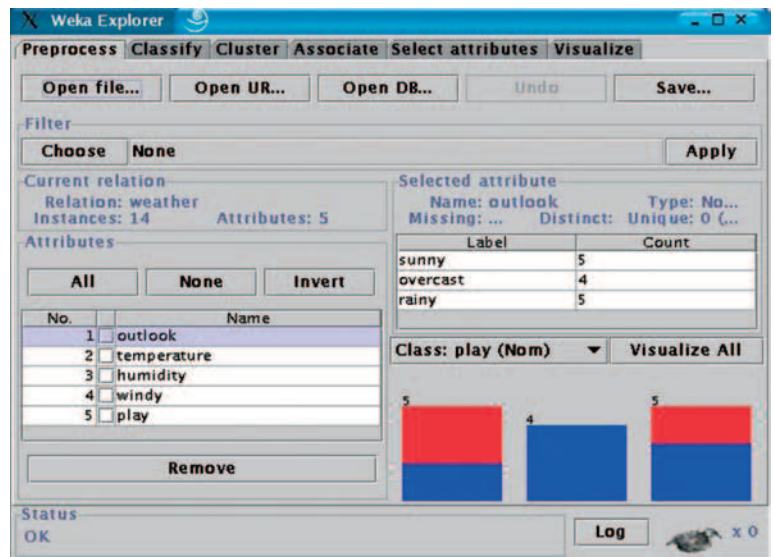
Anschließend präsentiert sich der „Weka GUI Chooser“, über welchen die eingangs erwähnten Applikationen erreichbar sind. Für den Einstieg in Weka eignet sich der Explorer, dessen Oberfläche in Abbildung 1 gezeigt ist.

Entdeckungsreise

Um die Datei *weather.arff* in den Explorer zu laden, wird auf der Seite „Preprocess“ die Schaltfläche OPEN FILE betätigt und die Datei aus dem Unterverzeichnis */data* ausgewählt. Ebenso könnten die Daten aus dem Internet oder einer Datenbank stammen. Auch hierfür stellt der Explorer entsprechende Schaltflächen zur Verfügung. Die Darstellung der Daten erfolgt in einer zweigeteilten Sicht. Links sind die Attribute mit ihrem Index und ihrem Namen aufgeführt. Darüber ist die Anzahl der Instanzen und die der Attribute sowie der Name der Relation sichtbar.

Auf der rechten Seite werden Informationen zu dem aktuell ausgewähltem Attribut dargestellt. Für numerische Attribute werden in der oberen Hälfte der Mittelwert und die Standardabweichung über die Attributwerte sämtlicher Instanzen ausgegeben. Bei nominalen Attributen werden hingegen die möglichen Werte mit ihrer jeweiligen Häufigkeit bezogen auf die Instanzen gelistet. Darunter ist die Verteilung der Werte hinsichtlich der vorher-

Abb. 1: Die Datei *weather.arff* im Explorer.



zusagenden Klasse graphisch abgebildet. Das Attribut, welches die Klasse bestimmt, kann über die darüber befindliche Auswahl Schaltfläche gesetzt werden.

Ein Wechsel auf die Seite „Classify“ offenbart schließlich die Steuerzentrale für die Klassifikation. Per Standardeinstellung ist als Klassifizierer „ZeroR“ gewählt und die zehnfache Kreuzvalidierung aktiviert. Zudem ist als Klassenattribut das letzte Attribut eingestellt, welches im Fall der Datei *weather.arff* tatsächlich jenes Attribut ist, dessen Wert vorhergesagt werden soll. Ein Klick auf die Schaltfläche START startet den Klassifizierer. Das Ergebnis der Klassifikation wird in dem rechten Textfeld nach kurzer Wartezeit eingeblendet. Gezeigt werden das erzeugte Modell des Klassifizierers und die Ergebnisse der Kreuzvalidierung auf diesem Modell.

Der Klassifizierer ZeroR hat nicht ohne Grund die Bezeichnung „Zero Rule“: Er erzeugt keinerlei Regeln auf Basis der Daten, sondern sagt für das Klassenattribut stets die Mehrheitsklasse voraus – in diesem Fall „yes“. Der in der Kreuzvalidierung ermittelte Wert für die Vorhersagegenauigkeit von etwa 64 Prozent ist somit auch ein guter Vergleichswert (allerdings nicht der einzige, welche die umfangreiche Ausgabe darbietet) für andere, komplexere Klassifizierer.

An Alternativen zu ZeroR mangelt es Weka nicht, wie ein Klick auf die Schaltfläche CHOOSE in der linken oberen Ecke offenbart. In einer Baumansicht werden die

verfügbaren Klassifizierer nach ihrer Funktionsweise gruppiert. In der Rubrik „trees“ (für die Klassifikation basierend auf Entscheidungsbäumen) findet sich beispielsweise der Klassifizierer „J48“. Nachdem dieser gewählt wurde, wird in dem Feld rechts von der Schaltfläche folgender Text angezeigt: „J48 -C 0.25 -M 2“. Dies sind die Parameter des Klassifizierers. Um sie zu ändern, reicht ein Klick auf das Textfeld. Es öffnet sich ein neues Fenster, in dem die Parameter justiert werden können. Über die Schaltfläche MORE wird eine Hilfe eingeblendet.

Für den hier beschriebenen Fall werden die Parameter auf ihren Standardeinstellungen belassen. Falls nötig können sie jedoch für eine Optimierung entsprechend angepasst werden. Auch dies kann in Weka mittels des Metaklassifizierers „CVPParameterSelection“ automatisiert werden. Metaklassifizierer führen selbst keine Klassifikation durch, sondern verwenden hierfür ein oder mehrere beliebig wählbare Klassifizierer, die in geeigneter Form kombiniert oder manipuliert werden. Ein Beispiel für die Anwendung eines solchen findet sich noch an späterer Stelle.

Die Klassifikation mit dem J48 bringt keine wesentliche Leistungssteigerung, obgleich er ein interessantes Modell erzeugt, dessen Darstellung als Entscheidungsbaum in Abbildung 2 zu sehen ist. Für dessen Anzeige muss in der „result list“ der entsprechende Eintrag mit der rechten Maustaste markiert und im anschließend eingeblen-

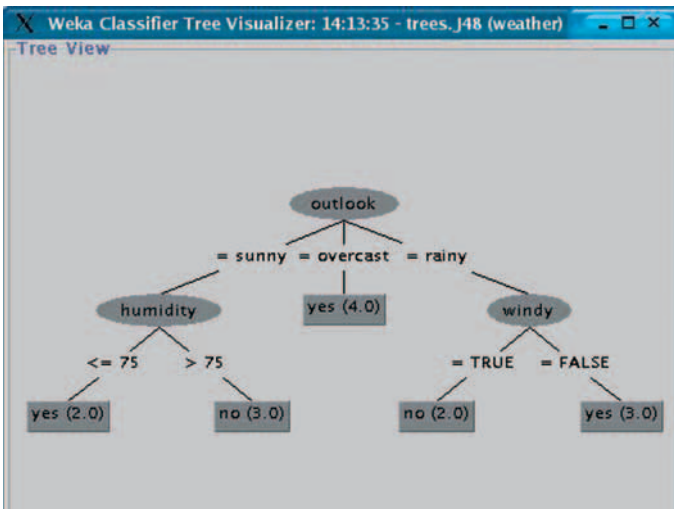


Abb. 2: Der von J48 erzeugte Entscheidungsbaum für die Wetterdaten

kommt, müssen in diesem speziellen Fall auch die Werte aus der Datei *weather.arff* diskretisiert werden, bevor der Klassifizierer zur Anwendung kommt. Für solche Aufgaben wird der Metaklassifizierer „FilteredClassifier“ benötigt. Anstelle von ID3 wird dieser dem Experiment hinzugefügt und seine Parameter werden mit dem Filter Discretize und dem Klassifizierer ID3 belegt.

Schließlich muss noch eine Datei (ARFF oder CSV) oder eine Datenbank bestimmt werden, in welche die Ergebnisse geschrieben werden. Um die aufwendige Konfigurationsprozedur bei einem erneuten Test nicht wiederholen zu müssen, können die Einstellungen als EXP-Datei gespeichert werden. Da hierfür die Java-Serialisierung Anwendung findet, können die Dateien jedoch nicht zwischen verschiedenen Installationen ausgetauscht werden.

Die Durchführung des Experiments wird über die Schaltfläche START auf der Seite „Run“ angestoßen. Sie braucht selbst auf schnellen Rechnern durchaus mehrere Minuten. Die untere Statusleiste gibt dabei über den Fortschritt Auskunft. Falls ein Rechnercluster zur Verfügung steht, können die Durchläufe oder alternativ die Datensätze auf verschiedene Rechner verteilt werden. Hierfür ist auf der Seite „Setup“ die Option „Advanced“ zu wählen und im Feld „Distribute Experiment“ die entsprechenden Einstellungen vorzunehmen. Auf den jeweiligen Rechnern muss hierzu das Programm „remoteEngine“ laufen und die notwendigen Sicherheitsrichtlinien müssen gesetzt sein. Die hierfür erforderlichen Schritte sind in einem Tutorial, welches unter [8] erhältlich ist, beschrieben.

Zahlenspiel

Nachdem das Experiment erfolgreich abgeschlossen wurde, kann auf die Seite „Analyse“ gewechselt werden und die Ergebnisse mittels der Schaltfläche EXPERIMENT importiert werden. Für den Fall, dass das Experiment fehlschlug, findet sich ebenfalls auf der Heft-CD die Datei *weather.result.arff*, welche mittels der Schaltfläche FILE eingelesen werden kann.

Per Standardeinstellung wird der statistische Test, wenn die Schaltfläche PERFORM TEST betätigt wird, auf dem prozentualen

deten Kontextmenü der Eintrag VISUALIZE TREE gewählt werden. Es handelt sich offensichtlich um ein Spiel, welches hauptsächlich bei schlechtem Wetter gespielt wird.

Experimentierfreude

Der Explorer macht es zwar leicht, verschiedene Klassifizierer ad hoc auf einer Datenmenge auszuführen, doch eine statistisch signifikante Aussage, ob ein Klassifizierer im Wettbewerb zu anderen besser oder schlechter ist, lässt sich durch einen bloßen Zahlenvergleich einer Messgröße wie die der Vorhersagegenauigkeit nicht herleiten. Vielmehr sind statistische Tests erforderlich, die unter einem gegebenen Signifikanzniveau beurteilen, ob ein Klassifizierer bezogen auf einen Vergleichswert tatsächlich besser abschneidet.

Für diese Aufgabe ist der Experimentier die geeignete Applikation. Er implementiert einen so genannten „t-Test“ (mehr hierüber findet sich in den meisten Statistikbüchern). Über seine Benutzeroberfläche, genauer auf der Seite „Setup“, können verschiedene Klassifizierer ausgewählt wer-

den, die auf ein oder mehreren Datensätzen evaluiert werden. Für dieses Experiment, bei dem eine Leave-One-Out Kreuzvalidierung Anwendung findet, werden folgende Klassifizierer verwendet: „ZeroR“, „JRip“ (ein regelbasierter Klassifizierer), „J48“, „SMO“ (eine Support-Vektor-Machine), „NaiveBayes“ und zu guter Letzt „ID3“ (ebenfalls aus der Rubrik „trees“).

Nicht immer ist der Klassifizierer an schlechten Ergebnissen schuld. Womöglich ist auch die gewählte Darstellungsform der Daten für die Problemstellung nicht geeignet. Um dies zu untersuchen, werden dem Experiment statt nur einer, zwei Datenmengen hinzugefügt. Ebenfalls im Unterverzeichnis /data findet sich eine Datei *weather.nominal.arff*, die anstelle der numerischen Attribute gleichbedeutende nominale Attribute besitzt. Die Umwandlung numerischer in nominale Werte, als Diskretisierung bezeichnet, erfolgt in Weka mit Hilfe des Filters „Discretize“.

Da der Klassifizierer ID3 ausschließlich mit nominalen Attributen zurecht-

Listing 2

ZeroR im Vergleich zu anderen Klassifizierern

Dataset	(1) rules.Ze	(2) rules	(3) trees	(4) funct	(5) bayes	(6) meta.
weather	(280) 64.29	63.21	64.29	43.57	64.29	64.29
weather.symbolic	(280) 64.29	69.64	50	62.86	50	78.57
	(v/*)	(0/2/0)	(0/2/0)	(0/2/0)	(0/2/0)	(0/2/0)

Anteil an korrekten Vorhersagen mit einem Signifikanzniveau von 0,05 ausgeführt. Dies bedeutet, dass, sollte ein Klassifizierer gemäß des Tests besser oder schlechter als der für den Vergleich ausgewählte Klassifizierer abschneiden, dies mit mindestens 95-prozentiger Wahrscheinlichkeit tatsächlich der Fall ist.

Der Vergleich des Klassifizierers ZeroR mit den übrigen Klassifizierern zeigt auf diesem Niveau keine Unterschiede. Dies lässt sich aus der in Listing 2 dargestellten Matrix ablesen. Die Spalten repräsentieren die Klassifizierer, wobei in der ersten Spalte die Werte des für den Vergleich ausgewählten Klassifizierers stehen. In den Zeilen werden die Datenmengen aufgelistet. Die Werte in den Zellen stehen für den Anteil an korrekten Vorhersagen des jeweiligen Klassifizierers für die jeweilige Datenmenge. Je nachdem, ob hinter dem Wert ein „v“, ein „*“ oder nichts steht, ist der Klassifizierer auf dieser Datenmenge signifikant besser, signifikant schlechter, oder weder signifikant besser noch signifikant

schlechter als der zu vergleichende Klassifizierer hinsichtlich des gewählten Vergleichswertes. Letzterer kann über die Auswahlchaltfläche COMPARISON FIELD geändert werden.

Ebenso kann der Klassifizierer gewechselt werden und zwar über die Schaltfläche SELECT BASE. Anstelle eines Klassifizierers kann hier auch der Eintrag „Ranking“ gewählt werden. Dabei treten die Klassifizierer wie in einem Fußballturnier gegenein-

Zahlreiche und vielfältige Methoden

ander an und in einem Ranking wird gezählt, wie oft ein Klassifizierer besser oder schlechter gegenüber den anderen abgeschnitten hat. Bei einem Signifikanzniveau von 0,05 gewinnt ID3 mit einem einzigen Punkt – ein klarer Favorit ist er also nicht. Um zu erfahren, gegen welchen Klassifizierer er „gewonnen“ hat, muss als Testbasis „Summary“ gewählt werden. Die Performance der beiden Datensätze kann verglichen werden, indem die unter „Row Key Fields“ und „Column Key Fields“ selektierten Felder vertauscht werden. Allerdings zeigt sich auch hier kein wesentlicher Unterschied.

Abschließend bleibt nun die Frage zu beantworten, weshalb ein komplexer Klassifizierer wie ID3 oder J48 gegenüber einem einfachen Verfahren wie dem der Wahl der Mehrheitsklasse durch ZeroR nicht wesentlich besser abschneidet. Mehrere Gründe sind hier denkbar: Erstens, die Daten folgen keinem Muster, das durch einen Entscheidungsbaum abgebildet werden könnte. In diesem Fall sollte jedoch zumindest einer der übrigen Klassifizierer auf einem der beiden Datensätze bessere Ergebnisse liefern. Zweitens, die Daten sind völlig zufällig, sie entsprechen also gar keinem Muster. Hiervon ist allerdings nicht auszugehen, da es sich schließlich um Hand erstellte Datensätze handelt. Drittens und letztens, die Datenbasis ist zu klein als dass hieraus brauchbare Modelle generiert werden könnten. Da die Datenmenge gerade einmal 14 Instanzen umfasst, ist dies wohl der wahrscheinlichste Fall.

Was übrig bleibt

Dieser Beitrag ist weder als vollständige noch detaillierte Einführung in das Maschinelle Lernen zu verstehen. Denn die Methoden des Data Minings sind ebenso zahlreich und vielfältig wie dessen Anwendungsgebiete: Sei es die Analyse des Kaufverhaltens von Kunden (wie sie beispielsweise bei Amazon betrieben wird), sei es die automatische Klassifikation von Dokumenten (wie zum Beispiel E-Mails) oder die Untersuchung der Genregulation (unter anderem zur Erkennung von Krebs), maschinelles Lernen kann überhaupt dort helfen, wo der Mensch den Überblick über seine Daten verloren hat. Das passende Werkzeug für diese Aufgabe haben Sie mit der umfangreichen und erweiterbaren Weka-Workbench bereits kennen gelernt. Für Ihre zukünftigen Experimente wünsche ich Ihnen viel Erfolg und interessante, neue Einblicke in Ihre Daten.

Martin Szugat ist seit mehreren Jahren als freischaffender Fachautor im Bereich der Softwareentwicklung tätig. Seit drei Jahren studiert er Bioinformatik in einem gemeinsamen Bachelor-Studiengang der Technischen Universität München (TUM) und der Ludwig-Maximilians-Universität München (LMU). Zudem arbeitet er an der Lehr- und Forschungseinheit Bioinformatik der LMU (www.bio.ifl.lmu.de) als studentische Hilfskraft. Dort beschäftigt er sich u.a. mit der Anwendung von Methoden des maschinellen Lernens im Bereich der Proteinklassifikation. Sie erreichen ihn unter Martin.Szugat@GMX.net. ■

„Data Mining“ von Ian H. Witten und Eibe Frank

Eine ausführliche Einführung in das Thema „Maschinelles Lernen“ bieten die Weka-Entwickler, Ian H. Witten und Eibe Frank, mit ihrem Buch „Data Mining“, in der deutschen Übersetzung beim Hanser Verlag erschienen. Das Buch behandelt die wichtigsten Verfahren aus den Bereichen Klassifikation, Clustering und Attributauswahl sowohl aus theoretischer als auch aus praktischer Sicht. Einzig die Black-Box-Verfahren, wie neuronale Netze und genetische Algorithmen, finden keine Erwähnung. Dagegen ist der Auswertung von Lernverfahren ein eigenes Kapitel gewidmet ebenso wie der Datenaufbereitung. Zu guter Letzt bietet das Buch auch eine Einführung in Weka – sowohl für Anwender als auch für Entwickler. Fazit: Für einen Einstieg in diese doch überaus komplizierte Materie ist das Buch uneingeschränkt empfehlenswert. Zudem bietet es – dank seines engen Bezuges zur Weka-Workbench – dem Leser die Möglichkeit, sein gelerntes Wissen sogleich in die praktische Erfahrung umzusetzen.

Ian H. Witten, Eibe Frank

Data Mining

379 Seiten, 49,90 Euro (laut Amazon.de)

Hanser Fachbuch, 2001

ISBN 3446215336

literatur

- [1] Weka: www.cs.waikato.ac.nz/~ml/weka/
- [2] Ian H. Witten und Eibe Frank: Data Mining: Practical Machine Learning Tools with Java Implementations. Morgan Kaufmann, 2000
- [3] Weka Dokumentation: www.cs.waikato.ac.nz/~ml/weka/doc_gui/index.html
- [4] Probekapitel aus „Data Mining“ (in Englisch): prdownloads.sourceforge.net/weka/Tutorial.pdf
- [5] Dokumentation zum ARFF-Format: www.cs.waikato.ac.nz/~ml/weka/arff.html
- [6] MARFF: www.cs.bris.ac.uk/~farrand/marff/
- [7] Weka-Parallel: weka-parallel.sourceforge.net/
- [8] Tutorial zum Experimentier: prdownloads.sourceforge.net/weka/Experiments.pdf